Grundlagen: Ausgabe auf die Konsole

Um dem Nutzer eines Programms Ergebnisse anzuzeigen oder über den Programmverlauf zu informieren, können Ausgaben auf die Konsole gegeben werden.

Was ist die Konsole?

Die Konsole oder auch Command Window ist ein Programm, welches ermöglicht, textbasiert mit dem Betriebssystem zu kommunizieren, um beispielsweise Programme zu starten. Mithilfe geeigneter Funktionen können wir die Konsole nutzen, um hier Text auszugeben.

C-Bibliotheken

C-Programme bestehen nicht nur aus selbstgeschriebenem Code, sondern können auf sogenannte **Bibliotheken** (Engl. *libraries*) zugreifen. Eine Bibliothek enthält bereits fertige Funktionen, die häufig benötigte Aufgaben übernehmen, z.B. mathematische Berechnungen, Speicherverwaltung oder – was wir heute benötigen – Ein- und Ausgaben auf die Konsole. Die zwei wohl wichtigsten C-Bibliotheken sind:

- **`stdio.h`** (Standard Input/Output) enthält Funktionen zur Ein- und Ausgabe über Konsole oder Dateien – z. B. `printf()` und `scanf()`.
- **`stdlib.h`** (Standard Library) enthält unter anderem Funktionen zur Speicherverwaltung, Typumwandlung oder Schlüsselwörter wie `EXIT_SUCCESS` und `EXIT_FAILURE`.

Um eine Bibliothek in ein C-Programm einzubinden, wird **ganz am Anfang** der C-Datei das Schlüsselwort **`#include`** gefolgt von dem Namen der Bibliothek in spitzen Klammern (`<...>`) geschrieben:

```
#include <stdio.h>
#include <stdlib.h>
 ...
```

Damit stehen alle Funktionen dieser Bibliotheken im Programm zur Verfügung. Das heißt umgekehrt, dass das Programm immer abstürzen wird, wenn man z.B. versucht, Text auf die Konsole auszugeben, ohne `stdio.h` inkludiert zu haben.

Die Funktion printf

Die Funktion **printf** aus der Standard-Ein- und Ausgabebibliothek **stdio.h** ermöglicht eine formatierte Ausgabe auf die Konsole. Mit ihr können sowohl einfacher Text als auch Werte von Variablen ausgegeben werden.

```
printf("Text", Parameter);
```

Der (optionale) Parameter in `printf()` erlaubt es, Werte (von Variablen) in den Text einzufügen. Dafür wird im Text eine Art **Platzhalter** (sogenanntes Formatzeichen) verwendet, die den Typ des Werts angibt:

```
int alter = 25;
printf("Ich bin %d Jahre alt.", alter);
```

Ausgabe:

```
Ich bin 25 Jahre alt.
```

Hinweis: Das Formatzeichen `%d` gibt an, dass es sich um eine Zahl (Engl. *digit*) handelt.
Siehe Tabelle unten für eine Liste von Formaten. Die Zeile `int alter = 25;` definiert eine Variable vom Typ Ganzzahl (Engl. *Integer*) mit dem Wert 25. Mehr zu Variablen in einem späteren Tutorium!

Formatierungsoptionen bei der Ausgabe

Bei `printf()` können die Ausgaben zusätzlich **formatiert** werden. So lässt sich etwa festlegen, wie viele Nachkommastellen angezeigt werden, wie breit die Ausgabe sein soll oder ob Nullen oder Leerzeichen aufgefüllt werden. Dazu verwendet man bestimmte Formatierungsparameter, die **innerhalb des Format-Strings** stehen.

```
printf("%[Flag][Breite][.Genauigkeit][Typvorsatz]Format", Parameter);
```

Flag	
-	Ausgabe linksbündig
+	Ausgabe rechtsbündig
Leerzeichen	Es wird einer positiven Zahl ein Leerzeichen vorangestellt
#	Auswirkung ist abhängig von Formattype
0	Verbleibende Stellen werden mit Nullen aufgefüllt

Typvorsatz	
l	Formattyp ist ein long
L	Formattyp ist ein doublelong

Format	
x,X	hexadezimal ohne Vorzeichen
lf	double; dezimal, Ausgabe von Nachkommastellen abhängig von Genauigkeit
f	float; Gleitkommazahl
d,i	int; dezimal mit Vorzeichen

Breite: Gibt die minimale Anzahl der auszugebenden Zeichen an. Wenn die Zahl kürzer ist, werden Leerzeichen (oder bei Flag `0` Nullen) ergänzt.
Genauigkeit: Gibt die Anzahl der Nachkommastellen bei Fließkommazahlen an.

Beispiel:

```
printf("%f\n", 14.367845);
printf("%.2f\n", 14.367845);
printf("%7.2f\n", 14.367845);
printf("%07.2f\n", 14.367845);
```

Ausgabe:

```
14.367845
14.37
  14.37
0014.37
```

Für eine genauere Beschreibung der einzelnen Parameter und Optionen zur Formatierung wird auf die Vorlesung und auf das dazugehörige Skript (Kapitel 2.6.1) verwiesen.

Sonderzeichen in Ausgaben

Bei der Ausgabe von Text können spezielle Zeichen verwendet werden, um z. B. einen Zeilenumbruch oder Tabulator einzufügen. Die wichtigsten sind in folgender Tabelle aufgelistet:

\n	newline (Zeilenumbruch)
\t	TAB (horizontaler Tabulator)
\r	carriage return (zurück zu Zeilenbeginn)
\b	backspace (ein Zeichen zurück)

Beispiel 1

Eine Gleitkommazahl (float) soll auf zwei Nachkommastellen gerundet ausgegeben werden. Für die Ausgabe sollen sechs Stellen freigehalten werden, leere Stellen werden mit Nullen befüllt. Der Code hierfür sieht folgendermaßen aus:

```
float x = 3.14159;
printf("Wert der Variable x = %06.2f", x);
```

Ausgabe:

```
Wert der Variable = 003.14
```

Beispiel 2

Eine Zeichenkette („String“ = Text) soll ausgegeben werden. Nach einem extra Absatz soll eine zusammengesetzte Ausgabe aus zwei Zeichenketten ausgegeben werden. Der Code hierfür sieht folgendermaßen aus:

```
printf("Das meistverkaufte Passagierflugzeug?\n\n");
printf("%s %s\n", "Airbus", "A320");
```

Ausgabe:

```
Das meistverkaufte Passagierflugzeug?

Airbus A320
```

Die Ausgabe von Zeichenketten kann sowohl als Parameter mit `%s` oder direkt innerhalb der Anführungsstriche erfolgen. Dieses Beispiel zeigt auch, dass mehrere Parameter in einer Ausgabe benutzt werden können. Dann entspricht die Reihenfolge der Parameter der der Formatzeichen.

Grundlagen: Eingabe über die Konsole

Damit der Nutzer aktiv mit dem Programm interagieren kann, werden Eingaben über die Konsole verwendet. So kann der Programmablauf von außen beeinflusst werden. Die Eingabe kann anschließend in einer Variable gespeichert werden.

Die Funktion scanf

Die Funktion **scanf** aus der Standard-Ein- und Ausgabebibliothek **stdio.h** ermöglicht eine formatierte Eingabe über die Konsole. Die Syntax ist sehr ähnlich zur Ausgabe mit **printf**.

```
scanf("%[Format]", Speicherort);
```

Format	
d	Ganzzahl im Dezimalsystem
i	Ganzzahl (integer)
u	positive Ganzzahl
p	Pointer
c	Einzelzeichen (character)
s	Zeichenkette (string)
lf	Gleitkommazahl (double)
e,E	Exponentialschreibweise
o	Ganzzahl im Oktalsystem
x,X	Ganzzahl im Hexadezimalsystem

Beispiel:

```
float x = 0;
scanf("%f", &x);
```

Die Variable x hat nun den Zahlenwert, der in die Konsole eingegeben wird.

Achtung! Der Speicherort (Variablenname), der in scanf angegeben wird, muss mit einem & versehen werden. Ausnahme bilden Strings. Genaueres dazu in einem späteren Tutorium.